A Partial Specification of an International Transmission Protocol

TO: E. Aupperle, R. Kahn, A. McKenzie, R. Metcalfe, R. Scantlebury, D. Walden, and H. Zimmerman

FROM: V. Cerf

SUBJECT: First pass draft of International Transmission Protocol

Please feel free to make comments, revisions, additions, and deletions from the corpus. I have tried to faithfully record my understanding of the protocol. From time to time, I have departed from the proposed terminology, but I believe I have done this only where it seems appropriate.

Your early response is critical. The next meeting is most likely to be scheduled for September 16 in Sussex, just after the NATO Advanced Institute, and this protocol will be a major topic for discussion.

Send replies to me at:

ERL 407
Stanford University
Stanford, California 94305
U. S. A.

Respectfully,

Vint Cerf

1973

TERMS OF REFERENCE

Connexion

It is imagined that between pairs of HOST computers, a single logical connexion is established upon which is multiplexed all communications between these two HOSTs. A connexion is a full-duplex channel, but it will be convenient to think of it as a pair of simplex, one-directional channels, each with a source and a destination.

Element

An element is a fixed length string of bits out of which messages (see below) of a communication between HOSTs are built. The size of an element is 8 bits by default unless it is changed explicitly by the sender.

Gateway

A gateway is the interface between networks. It may, in fact, not be a real entity (e.g., computer), but may merely be realized in software which lies on either side of a physical (optical, electromagnetic, acoustic, etc.) connection between the packet switches (or nodes) of two distinct networks.

HOST

A HOST is typically a computer which is attached to one or more packet switches (or nodes) in a constituent network. The HOST is a source and destination for messages. It may be the case that the HOST

is coexistent with a packet switch (e.g. the TIP in the ARPA network), or that it is in fact a terminal (e.g. RCP). In the sequel, it will be necessary to assume that a terminal cannot be a HOST unless it is capable of performing according to the specified protocol, and furthermore, the name (identification) of the terminal must be known to the packet switching subnet in the same way a serving HOST's identification is known. Note that a HOST may be connected to more than one packet switch (in different networks or the same network).

Message

A message is a well defined, formatted string of bits which has the property that it can be inserted into a constituent network at one packet switch where it will leave the network in the same form it entered. If the message is broken up into packets by the network this is invisible and the packets are reassembled before they leave the network. In the ARPA network, a message is at most about 8095 bits long. In the NPL network, this length is more like 257 bytes, and in RCP it is 64 bytes. Thus, a message is the largest bit string which can be entered into the network and retrieved from it without apparent change, breakage, etc. For some networks, messages and packets (see below) are identical, except possibly for some control information which is tacked onto the message while it is in transit through the net.

Node

A node is the elemental unit of a packet switching system. The term node and packet switch are used interchangeably in this paper, but a node might also be construed to be an addressable entity which participates

in the movement of packets within the network. This permits us to consider loop systems in which a node forwards but does not store packets which pass through it, on line switched systems.

Packet Switch

A packet switch is a computer which stores and forwards packets to achieve communication between HOSTS in a packet switching network. For purposes of this discussion, a gateway may be looked upon as a HOST by a packet switch.

Transmission Control Program (TCP)

A program in a HOST computer which realizes the International Transmission protocol enabling a collection of messages from processes and terminals associated with one HOST to be multiplexed along a connexion to another HOST in the same or foreign network. The TCP does not address itself to the problem of interprocess or terminal/process or terminal/terminal communication. The job of the TCP is merely to take a stream of messages produced by one HOST and reproduce the stream at a foreign receiving HOST without change.

Introduction

Let us begin with the assumption that we want to interconnect several distinct, resource-sharing computer networks. Each of these networks connects together HOST computers whose resources can be shared among the users of the network. If we are to achieve a similar ability between HOSTS residing in different networks, we must find a way for a HOST in one network to reproduce, without alteration, a stream of messages originating from a HOST in another network.

This is a primitive but essential necessity, and the mechanism we devise to do this will be called the International Transmission Protocol. This HOST level protocol will be implemented through a HOST program called the Transmission Control Program (TCP).

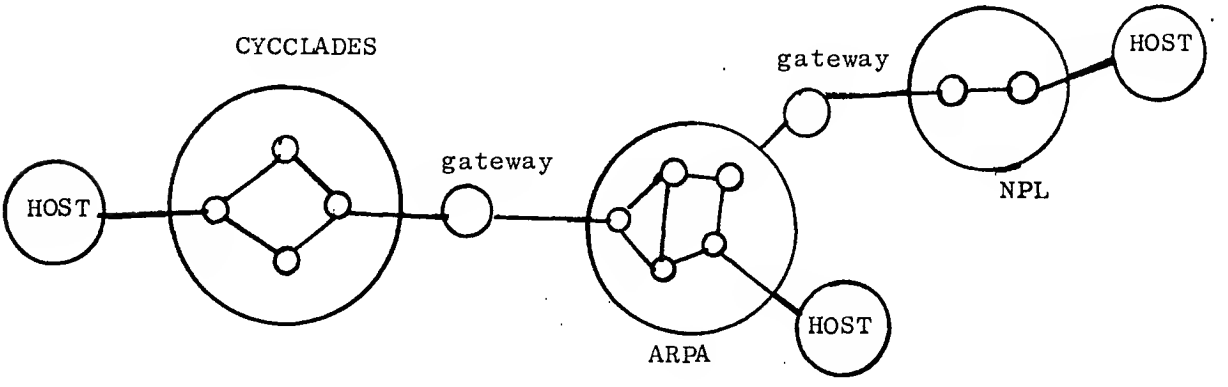
Consider a typical computer communication system. It is generally made up of nodes (packet switches, line switches, concentrators, multiplexors), and some communication medium connecting the nodes (telephone lines, wave guides, laser beams, radio or microwaves, satellite transponder, etc.).

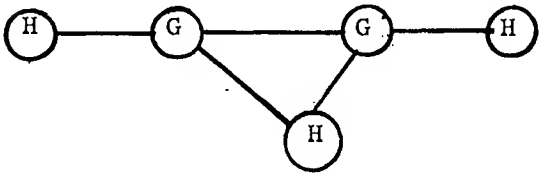
HOST computers are connected to the nodes in some fashion, and messages from one HOST to another are passed through the network.

The interconnection of two networks can be achieved through an interface or gateway which is connected to the nodes of two or more distinct networks (conceptually, it might be enough if two nodes were connected directly, but had software making each node regard the other as a HOST on its own network).

The gateway must accept messages from one network and pass them to another, reformatting the message as necessary for transmission in the new network.

If we treat the computer communication system of each network as a complicated communication line joining HOSTs and Gateway, then the Gateways appear to be international nodes, joined by two or more networks, facilitating communication between HOSTs.





The international transmission protocol (ITP) described herein is intended to:

1. Be resistant to failures in intermediate constituent networks and gateways.
2. Be unaffected by the maximum message sizes permitted in constituent networks and by intra- and inter-network timing idiosyncracies.

3. Be capable of exactly reconstituting a message stream originating in a foreign HOST.
4. Provide for high bandwidth and/or low delay transmission.
5. Require no status information in gateways.
6. Be relatively easy to implement.
7. Permit the receiving HOST to control the flow from the sending HOST.

Key Issues and Gratuitous Comments

A fundamental assumption behind the ITP is that message sizes are not uniform throughout all networks which are likely to be connected. It is tempting to try to pick a message size which all networks can transmit without alteration (e. g. 255 bytes), but our guess is that such an attempt is unlikely to succeed.

The inherent variability in message size accounts for much of the rigidity of the ITP. In another, as yet unpublished, position paper, R. Kahn proposes a scheme which appears more flexible. Kahn's design is based on a Simple Message Switched Protocol (reference Walden and Bressler), and solves the variable message size problem with a different message identification scheme than the one proposed for ITP. It is still not clear which of the two strategies, ITP or SMSP, is simplest.

The International Transmission Protocol

Message Format

Messages are made up of a fixed length header, containing control and addressing information, an integral number of fixed length elements containing the text of the message, and space for an optional checksum (see figure 2).

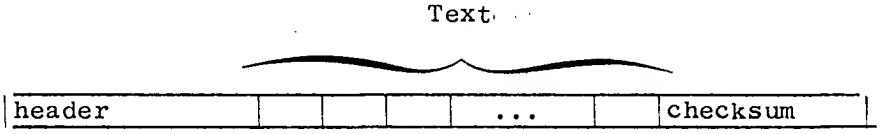


Figure 2

Message Format

We will take up the issue of variable element size later, but for the present we will assume a default size of 8 bits/element.

The ITP is resistant to failures in intermediate nodes and gateways. This is accomplished by arranging for positive acknowledgement of receipt of message by the receiving HOST. Furthermore, sending HOSTs are expected to time-out and retransmit messages which have not been acknowledged.

Any time-out and retransmission scheme requires that the receiver have the ability to detect duplicate transmissions. The ITP is no exception. Each message header carries a unique (short term) identifier which will be duplicated if the message is retransmitted.

The form that the unique identifier must take is determined in part by the fact that messages crossing through a gateway may require breaking into more than one message which may not be reassembled before delivery to a HOST. How can we uniquely identify the pieces of the broken message?

One solution to this problem is to uniquely label each element of the message, tacitly assuming that no gateway will break up a message into pieces smaller than one element, nor will messages be broken other than at element boundaries.

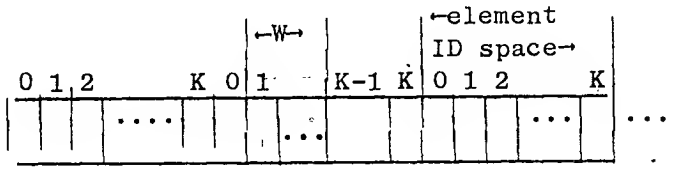


Figure 3

Bit stream, parsed into frames of k+1 elements.

W is the open window size.

In figure 3, we see a bit stream which has been parsed into a uniform, unbounded sequence of elements. Each element has an associated identifier which ranges from 0 to K. If we define frame size to be K+1 bits, then the ID of any given element is just its left bit ordinal modulo the frame size.

In order to permit a receiver to control the total traffic from a sender, the sender is constrained to have no more than W bits outstanding (unacknowledged) at one time. Initially, $W = \lfloor K/2 \rfloor$, but can be reduced by the receiver. Note that W cannot exceed $\lfloor K/2 \rfloor$ for reasons which will be apparent later.

A message can be made up of several sequential elements, and it is sufficient to identify the leftmost of these, while giving a count of the total number of bits in the message. Thus, the header format begins to look like Figure 5.

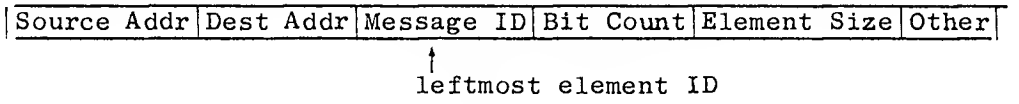
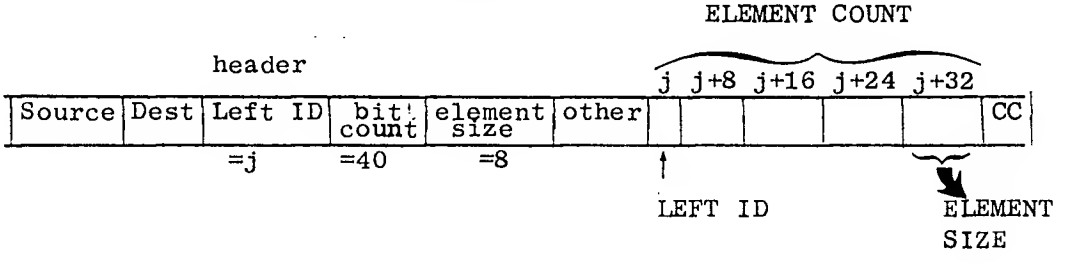


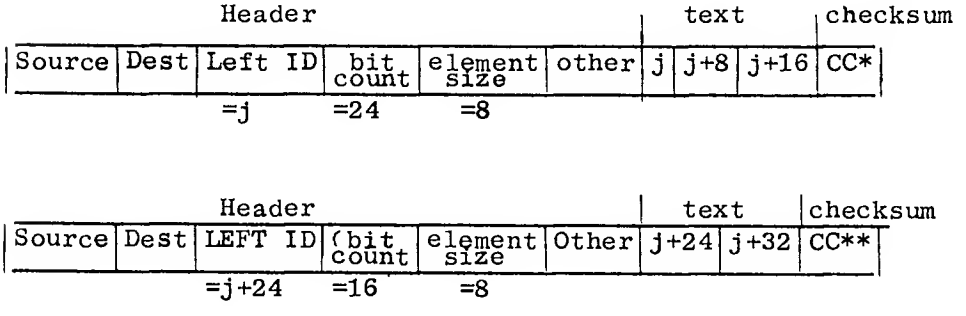
Figure 5

Header Format

If a message arrives at a gateway which finds that the message must be broken into pieces for transmission, each piece will have the same format. Since the text of each piece is constrained to consist of an integral number of elements, we can easily assign the correct unique identity to each element (see figure 6).



a. Original Message



b. Two new shorter messages

Figure 6

Message Breakage

Clearly, it is necessary that all networks have the capability of sending a message consisting of the header, checksum, and one element without breakage. Note that the element size (in bits) is carried in the header to facilitate gateway message reformatting.

Retransmission, positive acknowledgment and duplicate detection

Earlier, we alluded to a window size, W, which represented the maximum number of bits the sender was allowed to have outstanding (unacknowledged) by the receiver. Initially both sender and receiver have the left edge of the window positioned on bit 0 and the right edge on bit W-1. ($W \leq \lfloor K/2 \rfloor$).

The receiver uses the window to detect duplicate transmissions. When a message arrives, its left element ID is compared with the left edge of the window. If these match, the window is advanced to the right as many bits as the bit count indicates. As an acknowledgment the receiver returns to the sender the current left edge of the window. If the incoming message ID does not coincide with the present left edge of the window, the receiver determines whether the elements received lie inside or outside the window. If inside, the receiver can either ignore the elements, or mark them received, but not acknowledged. No acknowledgment will be sent for these elements until the missing elements to the left have all been successfully received. However, if the receiver sends as an acknowledgment the current left window edge, this may serve to stimulate retransmission of the missing elements. If the elements lie outside the window, then they are discarded, and an acknowledgment bearing the current left window edge ID is sent to the sender.

In fact, these acknowledgments are sent "piggyback" with other messages traveling to the source using a new field in the header labeled ACK (see figures 6 and 7). If no messages are awaiting transmission, then an empty message is sent.

Source	Destination	Left Element ID	Bit Count	Element size	Left ID ACK	Other

HEADER FORMAT

Figure 7

The sender transmits a group of elements as a message and starts a time-out. If the sender times out before an acknowledgment for the elements is received, the sender retransmits the elements. In the simplest implementation, out-of-order acknowledgments are ignored (ultimately causing retransmission). In the more complex implementation, they are accepted and the associated elements marked as successfully acknowledged. The sender does not advance the window to the right unless it receives acknowledgments for elements lying consecutively to the right of the left window edge.

One can easily be convinced that the sender can never advance the left edge of his window beyond the left edge of the receiver's window (flow control), and barring total disaster, all elements are eventually acknowledged, even if they require retransmission. Furthermore, the receiver has no trouble distinguishing between duplicate and original elements.

There are several important points to notice. First, the receiver can actually advance his window until its left edge is W bits ahead of the sender (all elements received, but all acknowledgments lost). Since

the sender will ultimately retransmit to prompt the receiver again for acknowledgments, it is important that the receiver not confuse a retransmitted element for an original one. If $W > \lfloor K/2 \rfloor$, where K+1 is the frame size, then such confusion can occur, hence the restriction $W \leq \lfloor K/2 \rfloor$.

The second point to notice is that a range of implementations is possible, ranging from purely sequential receipt and acknowledgment to a more complex implementation allowing non-sequential receipt and acknowledgment.

The third point is that negative acknowledgments are accomplished implicitly if the receiver acknowledges only the left window edge on receipt of elements within but not contiguous with the left window edge. If the sender keeps pointers to current left edge, last sent, and right window edge elements, then each acknowledge which fails to advance the sender's left window edge can be treated as a negative acknowledgment.

Flow Control and Element Size.

Initially, the communication channel between TCP's is assumed to consist of 8 bit elements with a window of $W = \lfloor K/2 \rfloor$ bits where $K + 1$ is the frame size.

The receiver may want to change the window size (as may the sender), and either may want to change the element size.

Such changes must be done in a controlled manner, so we postulate the existence of control messages sent from one TCP to another. The message format, since it embeds control messages within the data stream, must signal the presence of a control message by a flag in the header. Figure 9 shows the message header format.

Source	Destination	Left Elem ID	bit count	Elem size	Ack	Type	Text	CC

Message Header

Figure 9

For the moment, only three types of messages are proposed.

a. Type 0 = regular message for a processor terminal

b. Type 1 = control message

c. Type 2 = status message

Suppose the receiver wants to tell the sender about a new window size W'. The receiver sends a control message "DO SETWINDOWSIZE W'." The sender will not respond until the window can be either reduced or expanded on the right to size W'. When this occurs, the sender responds with "WILL SETWINDOWSIZE W'." Notice that the window should not be expanded beyond W = $\lfloor K/2 \rfloor$ nor should it be reduced (right edge moved leftward) over previously, acknowledged elements.

Element Size

In an early design, it was assumed that all messages in the stream from one TCP to another would consist of messages with the same element size. This size could be changed by negotiation between TCP's. Once we realize that control messages must be embedded in the stream, it is made easier to imagine that every message carries information about element size, and is guaranteed to consist of an integral number of elements, but each message may have a different element size. This permits the multiplexing of messages between several processes with different length elements, but still gives a good flow control mechanism to the receiver. Since message ID's are at the bit level, out-of-order

message can still, with some work, be inserted in the correct place in the circular buffer whose length is W bits.

Since each message can have a different element size, there is no need to negotiate to change it.

Checksum

The checksum is optionally computed over the header and message text. If not computed, this field is all 1's, otherwise, it contains a folded checksum. If the originating TCP fails to put a checksum in (i.e. field is all 1's), then at some point at or before the first gateway, this checksum should be computed and put in its proper place.

If a message is broken up, the new pieces should have checksums generated by the gateway.

Messages which fail to pass the checksum test should be discarded (and negative acknowledge produced -- e.g. current and unmoved left edge of window).

Routing

Initially, the gateway destination should be selected by the TCP from a table of networks versus gateway addresses. The constituent network nodes may choose to re-route messages which are destined for highly congested areas.

Addressing

We take the position that only sufficient addressing need be provided to get a message to a particular TCP. The address field must

be broken into a subfield for network, and one for TCP identifier. (see figure 10). Although it is not mandatory, most large networks may prefer a hierarchical addressing structure

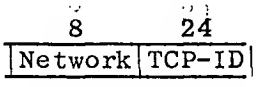


Figure 10
Source or Destination Address

in which the TCP-ID is broken into sections. This is purely up to each network.

PROBLEM AREAS

1. Setting time-outs for retransmission
2. Header field sizes:
   a. element size - 8 bits
   b. left ID - 16 bits
   c. bit count - 16 bits
   d. address - 32 bits
   e. checksum - 16 bits
   f. ACK - 16 bits
   g. TYPE - 8 bits
3. Keeping status table sizes down (Walden's Link Table idea).
4. Note: I went to bit addressing instead of element addressing because it became clear that I want to multiplex messages of many different sizes along the same TCP-TCP connexion.

5. Status information obtainable by TCP from distant TCP? (machine word size, window size, TCP performance data...).

6. Error messages?

7. How to propagate internal network status to HOSTs via ITP? (e.g. HOST DEAD MSG NOT DELIVERED).

8. Gateway Accounting?

9.

10.